World Academy of Science, Engineering and Technology
International Journal of Mathematical and Computational Sciences
Vol:2, No:4, 2008

Local Error Control in the RK5GL3 Method

J.S.C. Prentice

***Abstract*—**The RK5GL3 method is a numerical method for solving initial value problems in ordinary differential equations, and is based on a combination of a fifth-order Runge-Kutta method and 3-point Gauss-Legendre quadrature. In this paper we describe an effective local error control algorithm for RK5GL3, which uses local extrapolation with an eighth-order Runge-Kutta method in tandem with RK5GL3, and a Hermite interpolating polynomial for solution estimation at the Gauss-Legendre quadrature nodes.

***Keywords*—**RK5GL3, RKrGLm, Runge-Kutta, Gauss-Legendre, Hermite interpolating polynomial, initial value problem, local error.

Open Science Index, Mathematical and Computational Sciences Vol:2, No:4, 2008 publications.waset.org/8341/pdf

I. Introduction

The RKrGLm method [1] for solving

$$y' = f(x,y) \qquad y(x_0) = y_0 \qquad a \leqslant x \leqslant b \tag{1}$$

is based on an explicit Runge-Kutta of order r (RKr) and m-point Gauss-Legendre quadrature (GLm). The method has a global error of order $r+1$, which is the same order as the local order of the underlying RKr method. In this paper we consider the particular method RK5GL3, and describe an effective algorithm for controlling the local error in RK5GL3.

II. Terminology and relevant concepts

In this section we describe terminology and concepts relevant to the paper, including a brief description of the RKrGLm method.

A. Explicit Runge-Kutta methods

We denote an explicit RK method for solving (1) by

$$w_{i+1} = w_i + h_i F(x_i, w_i) \tag{2}$$

where $h_i \doteqdot x_{i+1} - x_i$ is a stepsize, w_i denotes the numerical approximation to $y(x_i)$, and $F(x,y)$ is a function associated with the particular RK method (indeed, $F(x,y)$ could be regarded as the function that *defines* the method).

B. Local and global errors

We define the global error in a numerical solution at x_i by

$$\Delta_i \doteqdot w_i - y_i, \tag{3}$$

and the local error at x_i by

$$\varepsilon_{i+1} \doteqdot [y_i + h_i F(x_i, y_i)] - y_{i+1} \tag{4}$$

In the above, y_i is the true solution at x_i. Note that the exact value y_i is used in the bracketed term in (4).

Note also that for the derivative $y' = f(x,y)$ we have

$$f(x_i, w_i) = f(x_i, y_i + \Delta_i) = f(x_i, y_i) + \Delta_i f_y(x_i, \vartheta_i) \tag{5}$$

where $\vartheta_i \in (y_i, y_i + \Delta_i)$, so that an error of Δ_i in w_i results in an error of $O(\Delta_i)$ in $f(x_i, w_i)$.

Justin Prentice is with the Department of Applied Mathematics, University of Johannesburg, South Africa, email: jprentice@uj.ac.za

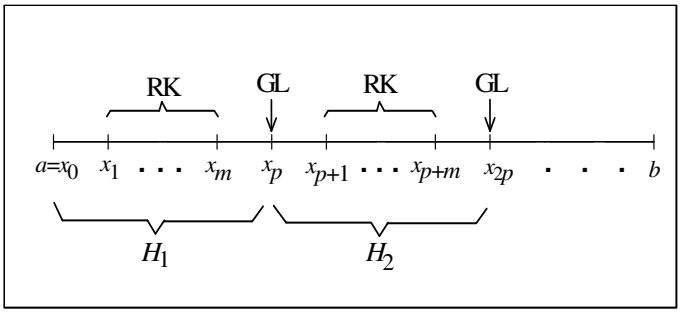


Fig. 1. RKGL algorithm for the first two subintervals H_1 and H_2 on $[a,b]$.

C. Gauss-Legendre quadrature

Gauss-Legendre quadrature on $[u,v]$ with m nodes is given by [2]

$$\int_u^v f(x,y)\,dx = h\sum_{i=1}^{m} C_i f(x_i, y_i) + O\left(h^{2m+1}\right) \tag{6}$$

where the nodes x_i are the roots of the Legendre polynomial of degree m on $[u,v]$. Here, h is the average separation of the nodes on $[u,v]$, a notation we will adopt from now on, and the C_i are appropriate weights. For GL3, the roots of the 3rd degree Legendre polynomial on $[-1,1]$ are

$$\begin{aligned} \widetilde{x}_1 &= -0.77459666924148 \\ \widetilde{x}_2 &= 0 \\ \widetilde{x}_3 &= 0.77459666924148 \end{aligned} \tag{7}$$

and are mapped to corresponding nodes x_i on $[u,v]$ via

$$x_i = \frac{1}{2}\left[(v-u)\,\widetilde{x}_i + u + v\right]. \tag{8}$$

Also, the average node separation on $[-1,1]$ is $1/2$, and so h on $[u,v]$ is given by

$$h = \frac{1}{2}\left(\frac{v-u}{2}\right), \tag{9}$$

while the weights

$$C_1 = \frac{10}{9}, \quad C_2 = \frac{16}{9}, \quad C_3 = \frac{10}{9} \tag{10}$$

are constants on any interval of integration.

D. The RKrGLm algorithm

We briefly describe the general RKrGLm algorithm on the interval $[a,b]$, with reference to Figure 1.

Subdivide $[a,b]$ into N subintervals H_i. At the RK nodes we use RKr :

$$w_{i+1} = w_i + h_i F(x_i, w_i) \tag{11}$$

At the GL nodes we use m-point GL quadrature:

$$w_{2p} = w_p + h\sum_{i=1}^{m} C_i f(x_i, w_i) \tag{12}$$

Note that $p \doteqdot m+1$.

The GL component is motivated by

$$\int_{x_p}^{x_{2p}} f(x,y)\,dx = y_{2p} - y_p \approx h\sum_{i=1}^{m} C_i f(x_i, y_i) \tag{13}$$

$$\Rightarrow y_{2p} \approx y_p + h\sum_{i=1}^{m} C_i f(x_i, y_i). \tag{14}$$

Of course, in RK5GL3 we have $r = 5, m = 3$ and $p = 4$ in the above. The RKrGLm algorithm has been shown to be consistent, convergent and zero-stable [1].

E. Local error at the GL nodes

The local error at the GL nodes is defined in a similar way to that for a one-step method:

$$\int_{x_p}^{x_{2p}} f(x,y)\,dx = \underbrace{y_{p+m+1}}_{2p} - y_p \tag{15}$$

$$= h\sum_{i=1}^{m} C_i f(x_i, y_i) + O\left(h^{2m+1}\right) \tag{16}$$

$$\Rightarrow \varepsilon_{2p} = \underbrace{\left[y_p + h\sum_{i=p+1}^{p+m} C_i f(x_i, y_i)\right]}_{\text{exact values of } y(x)} - y_{2p} \tag{17}$$

$$= O\left(h^{2m+1}\right). \tag{18}$$

Note that in the upper limit of integration $x_{2p} = x_{p+m+1}$. In RK5GL3, the local error at the GL nodes is $O\left(h^7\right)$.

F. Implementation of RK5GL3

There are a few points regarding the implementation of RK5GL3 that need to be discussed:

- If we merely sample the solutions at the GL nodes, treating the computations at the RK nodes as if they were the stages of an ordinary RK method, then RK5GL3 would be reduced to an inefficient one-step method. This is not the intention behind the development of RK5GL3; rather, RK5GL3 represents an attempt to improve the efficiency of RK5, simply by replacing the computation at every fourth node by a quadrature formula which does not require evaluation of any of the stages in the underlying RK5 method.
- Of course, it is clear from the above that on H_1 the RK nodes are required to be consistent with the nodes necessary for GL quadrature. If, however, the RK nodes are located differently (perhaps due to a local error control mechanism, for example) then it is a simple matter to construct a Hermite interpolating polynomial of degree seven (which has order eight error) using the solutions at the nodes $\{x_0, \ldots, x_3\}$. Then, assuming x_0 maps to -1 and x_3 maps to the Legendre polynomial root $\widetilde{x}_3$ on $[-1, 1]$, the position of the other nodes $\{x_1^*, x_2^*\}$ suitable for GL quadrature may be determined, and the Hermite polynomial may be used to find approximate solutions of order eight at these nodes, thus facilitating the GL component of RK5GL3. A similar process is carried out on the next subinterval H_2, and so on. Indeed, as will be seen, the Hermite polynomial discussed here will play an important part in our error control algorithm.

G. The Runge-Kutta methods used in the algorithm

The RK method used in RK5GL3 is an explicit fifth-order method due to Fehlberg [3], which we denote RK5. The explicit eighth-order method used as the tandem method for error estimation in our error control algorithm is also due to Fehlberg [4], [5], and is denoted RK8.

H. The Hermite interpolating polynomial

If the data $\{x_i, y_i, y_i' : i = 1, \ldots, m\}$ are available, then a polynomial $\mathcal{H}(x)$, of degree at most $2m-1$, with the interpolatory properties

$$\mathcal{H}(x_i) = y_i \qquad \mathcal{H}'(x_i) = y_i' \tag{19}$$

for each i, may be constructed. If the nodes x_i are distinct, then $\mathcal{H}(x)$ is unique. This approximating polynomial is known as the Hermite interpolating polynomial [6], and has an approximation error given by

$$y(x) - \mathcal{H}(x) = \frac{y^{(2m)}(\xi(x))}{(2m)!}\prod_{i=1}^{m}(x - x_i)^2 \tag{20}$$

where $x_1 < \xi(x) < x_m$. If h is the average separation of the nodes on $[x_1, x_m]$, it is possible to write $x - x_i = \sigma_i h$, where σ_i is a suitable constant, and hence

$$y(x) - \mathcal{H}(x) = O\left(h^{2m}\right). \tag{21}$$

The algorithm for determining the coefficients of $\mathcal{H}(x)$ is linear, as in

$$\mathbf{c} = \mathbf{A}^{-1}\mathbf{b} \tag{22}$$

where $\mathbf{c}$ is a vector of the coefficients of $\mathcal{H}(x)$, $\mathbf{A}$ is the relevant interpolation matrix, and $\mathbf{b}$ is a vector containing y_i and y_i'. The details of these terms need not concern us here; rather, if an error $O(\Delta)$ exists in each of y_i and y_i', then an error of $O(\Delta)$ will exist in each component of $\mathbf{c}$. Moreover, since $\mathcal{H}(x)$ is linear in its coefficients, then an error of $O(\Delta)$ will also exist in any computed value of $\mathcal{H}(x)$. Consequently, we may write

$$y(x) - \mathcal{H}(x) = O\left(h^{2m}\right) + O(\Delta) \tag{23}$$

where the $O(\Delta)$ term arises from errors in y_i and y_i'. We have assumed, of course, that the errors in y_i and y_i' are of the same order, which is the situation that we will encounter later.

Open Science Index, Mathematical and Computational Sciences Vol:2, No:4, 2008 publications.waset.org/8341/pdf

Open Science Index, Mathematical and Computational Sciences Vol:2, No:4, 2008 publications.waset.org/8341/pdf

III. Local error control in RK5GL3

A. The order of the tandem method

The idea behind the use of a *tandem* method is that it must be of sufficiently high order such that, relative to the approximate solution generated by RK5GL3, the tandem method yields a solution that may be assumed to be essentially exact. This solution is propagated in both RK5GL3 and the tandem method itself, and the difference between the two solutions is taken as an estimate of the local error in RK5GL3. This amounts to so-called *local extrapolation* and is not dissimilar in spirit to error estimation techniques employed using Runge-Kutta embedded pairs [7], [8].

To decide on an appropriate order for the tandem method we consider the local error at the GL nodes

$$\begin{aligned}\varepsilon_{2p} &= y_p + h\sum_{i=p+1}^{p+m} C_i f\left(x_i, y_i\right) - y_{2p} \\ &= \left[w_{p,t} - \Delta_{p,t} + h\sum_{i=p+1}^{p+m} C_i f\left(x_i, w_{i,t} - \Delta_{i,t}\right)\right] \\ &\quad - \left(w_{2p,t} - \Delta_{2p,t}\right) \qquad (24)\end{aligned}$$

where $w_{k,t}$ is the solution from the tandem method at x_k, and $\Delta_{k,t}$ is the global error in $w_{k,t}$. Expanding the term in the sum in a Taylor series gives

$$\begin{aligned}\varepsilon_{2p} &= w_{p,t} + h\sum_{i=p+1}^{p+m} C_i f\left(x_i, w_{i,t}\right) - w_{2p,t} \\ &\quad -\Delta_{p,t} + \Delta_{2p,t} \\ &\quad +h\sum_{i=p+1}^{p+m} C_i f_y\left(x_i, \zeta_{i,t}\right)\Delta_{i,t} \qquad (25)\end{aligned}$$

and so

$$\begin{aligned}&w_{p,t} + h\sum_{i=p+1}^{p+m} C_i f\left(x_i, w_{i,t}\right) - w_{2p,t} \\ &= \varepsilon_{2p} + \Delta_{p,t} - \Delta_{2p,t} \\ &\quad -h\sum_{i=p+1}^{p+m} C_i f_y\left(x_i, \zeta_{i,t}\right)\Delta_{i,t} \qquad (26)\end{aligned}$$

The sum on the rhs is of higher order than $\Delta_{p,t} - \Delta_{2p,t}$, because of the multiplication by h, and since we cannot expect, in general, that $\Delta_{p,t} - \Delta_{2p,t} = 0$, the term in parentheses must be $O\left(h^q\right)$, where q is the *global* order of the tandem method. Since $\varepsilon_{2p} = O\left(h^7\right)$ in the RK5GL3 method, we require $q > 7$ in order for

$$w_{p,t} + h\sum_{i=p+1}^{p+m} C_i f\left(x_i, w_{i,t}\right) - w_{2p,t} \approx \varepsilon_{2p} \qquad (27)$$

to be a good (and asymptotically $(h \to 0)$ correct) estimate for the local error in RK5GL3. The first two terms on the lhs of (27) arise from RK5GL3 with the tandem solution as input, while $w_{2p,t}$ is the tandem solution at x_{2p}.

The implication, then, is that the tandem method must have a global order of at least eight. Hence, we have chosen the Fehlberg method mentioned earlier (RK8) as the tandem method for use with RK5GL3. We remind the reader that RK5 and RK8 used here are independent, and are not an embedded pair. There may be practical reasons relating to efficiency that could suggest the use of a suitable embedded pair, but we will address this issue at a later stage. We also acknowledge that our choice of q differs from conventional wisdom (which would choose $q = 7$ so that the local order of the tandem method is eight), but it is clear from (26) that the propagation of the tandem solution requires the global order of the tandem method to be greater than the order of ε_{2p}. Of course, at the RK nodes the local order is six, so the tandem method RK8 is more than suitable at these nodes.

B. The error control algorithm

We describe the error control algorithm on the first subinterval $H_1 = [x_0(= a), x_4]$ (see figure 1). This process is repeated on subsequent subintervals.

Solutions $w_{1,5}$ and $w_{1,8}$ are obtained at x_1 using RK5 and RK8, respectively. We assume

$$|w_{1,5} - y_1| = L_1 h_0^6 \approx |w_{1,5} - w_{1,8}| \qquad (28)$$

where $h_0 \doteqdot x_1 - x_0$ and L_1 is a local error coefficient (we will discuss the choice of a value for h_0 later). The exponent of six indicates the order of the local error in RK5. We then demand that

$$\begin{aligned}\left|\frac{w_{1,5} - y_1}{y_1}\right| &\approx \left|\frac{w_{1,5} - w_{1,8}}{w_{1,8}}\right| \leqslant \varepsilon_R \qquad (29)\\ \Rightarrow |w_{1,5} - w_{1,8}| &\leqslant \varepsilon_R\,|w_{1,8}| \qquad (30)\end{aligned}$$

where ε_R is a user-defined tolerance. If this inequality is violated we assume we can find a new stepsize h_0^* such that

$$L_1\left(h_0^*\right)^6 = \varepsilon_R\,|w_{1,8}| \Rightarrow h_0^* = 0.9\left(\frac{\varepsilon_R\,|w_{1,8}|}{L_1}\right)^{\frac{1}{6}} \qquad (31)$$

and then find new solutions $w_{1,5}$ and $w_{1,8}$ using h_0^*. The factor 0.9 in (31) is a 'safety factor' allowing for the fact that $w_{1,8}$ is not truly exact. To cater for the possibility $w_{1,8} \approx 0$ we actually demand

$$|w_{1,5} - w_{1,8}| \leqslant \max\left\{\varepsilon_A, \varepsilon_R\,|w_{1,8}|\right\} \qquad (32)$$

where ε_A is a user-defined 'absolute' tolerance. We then set $h_1 = h_0^*$ and proceed to the node x_2, where the error control process is repeated, and similarly for x_3. The process of recalculating a solution using a new stepsize is known as a *step rejection*.

In the event that the condition in (32) is satisfied, we still calculate a new stepsize h_0^* (which would now be larger than h_0) and set $h_1 = h_0^*$, on the assumption that if h_0^* satisfies (32) at x_1, then it will do so at x_2 as well (however, we also place an upper limit on h_0^* of $2h_0$, although the choice of the factor two here is somewhat arbitrary). In the worst-case scenario we would find that h_1 is too large and a new, smaller value h_1^* must be used. The exception occurs when $|w_{1,5} - w_{1,8}| = 0$. In this case we simply set $h_1 = 2h_0$ and proceed to x_2.

The above is nothing more than well-known local relative error control in an explicit RK method using local extrapolation. It is at the GL node x_4 that the algorithm deviates from the norm.

Open Science Index, Mathematical and Computational Sciences Vol:2, No:4, 2008 publications.waset.org/8341/pdf

Once error control at x_1, x_2 and x_3 has been effected (which necessarily defines the positions of x_1, x_2 and x_3 due to stepsize modifications that may occur), the location of x_4 must be determined such that the local relative error at x_4 is less than $\max\{\varepsilon_A, \varepsilon_R |w_{4,8}|\}$. To this end, we utilize the map (8), demanding that $x_0\,(=u)$ corresponds to -1 and x_3 corresponds to the Legendre polynomial root $\widetilde{x}_3$ in (7). This allows $x_4\,(=v)$ to be found, where x_4 corresponds to 1, and new nodes x_1^* and x_2^* to be determined such that $\{x_1^*, x_2^*, x_3\}$ are consistent with the GL quadrature nodes on $[x_0, x_4]$. We seek to perform GL quadrature on $[x_0, x_4]$ using the nodes $\{x_1^*, x_2^*, x_3\}$; however, we do not have solutions $w_{1,8}^*$ and $w_{2,8}^*$ at x_1^* and x_2^*. Hence, we construct the Hermite interpolating polynomial $\mathcal{H}(x)$ on $[x_0, x_3]$ using the nodes $\{x_0, x_1, x_2, x_3\}$ and the solutions that have been obtained at these nodes; of course, the derivative of $y(x)$ at these nodes is given by $f(x, y)$. We use the eighth-order solutions that are available, so that we expect the approximation error in $\mathcal{H}(x)$ to be $O\left(h^8\right)$, as shown in (5) and (23). The solutions at x_1^* and x_2^* are then obtained from $\mathcal{H}(x_1^*)$ and $\mathcal{H}(x_2^*)$. GL quadrature then gives w_4 with local error $O\left(h^7\right)$, as per (17). The tandem method RK8 is used to find $w_{4,8}$, and $|w_4 - w_{4,8}|$ is then used for error control: we know that the local error in w_4 is $O\left(h^7\right)$, where h here is the average node separation on $[x_0, x_4]$; if the local error is too large then a new average node separation h^* is determined; using h^*, a new position for x_4, denoted x_4^*, is found from $x_4^* = x_0 + 4h^*$; if $x_4^* > x_3$, we redefine the nodes $\{x_1^*, x_2^*, x_3\}$, find eighth-order solutions at these new nodes using $\mathcal{H}(x)$, and then find solutions at x_4^* using GL quadrature and RK8; if $x_4^* \leqslant x_3$, we reject the GL step since there is now no point in finding a solution at x_4^*. After all this, the node x_4^* or x_3 (if $x_4^* \leqslant x_3$) defines the endpoint of the subinterval H_1; the stepsize h is set equal to the largest separation of the nodes on H_1, and the entire error control procedure is implemented on the next subinterval H_2. Note also that it is the eighth-order solution at the endpoint of H_1 that is propagated in the RK solution at the next node.

C. Initial stepsize

To find a stepsize h_0 to begin the calculation process, we assume that the local error coefficient $L_1 = 1$ and then find h_0 from

$$h_0 = \left(\max\{\varepsilon_A, \varepsilon_R |y_0|\}\right)^{\frac{1}{6}}. \tag{33}$$

Solutions obtained with RK5 and RK8 using this stepsize then enable a new, possibly larger, h_0 to be determined, and it is this new h_0 that is used to find the solutions $w_{1,5}$ and $w_{1,8}$ at the node x_1.

D. Final node

We keep track of the nodes that evolve from the stepsize adjustments, until the end of the interval of integration b has been exceeded. We then backtrack to the node on $[a, b]$ closest to b (call it x_{f-1}), determine the stepsize $h_{f-1} \doteqdot b - x_{f-1}$, and then find $w_{b,5}$ and $w_{b,8}$, the numerical solutions at b using RK5 and RK8, with h_{f-1}, x_{f-1} and $w_{f-1,8}$ as input for both RK5 and RK8. This completes the error control procedure.

IV. Comments on efficiency

Our intention has been to develop an effective local error control algorithm for RK5GL3, and we believe that the above-mentioned algorithm achieves this objective. However, we do acknowledge that our procedure is probably not as efficient as it could be. For example, it would be less computationally expensive to use an embedded RK pair instead of independent RK5 and RK8 methods. Such a pair, known as DOPRI853, does in fact exist, and is due to Dormand and Prince [9]. In our algorithm, RK5 is a six stage method, and RK8 is a 13 stage method, implying at least 19 stage evaluations at each RK node. DOPRI853, on the other hand, is a 12 stage method containing both fifth-order and eighth-order methods. This suggests a ratio of computational effort of 12/19=63%, so that using DOPRI853 might require only about two-thirds of the effort of the tandem algorithm. Regrettably, at the time of writing, DOPRI853 had not been tested in this error control capacity.

Nevertheless, we will show in the next section by way of two numerical examples that our error control algorithm is certainly an effective one.

V. Numerical examples

By way of example, we solve

$$y' = \frac{1}{1+x^2} - 2y^2 \tag{34}$$

on $[0, 5]$ with $y(0) = 0$, and

$$y' = \frac{y}{4}\left(1 - \frac{y}{20}\right) \tag{35}$$

on $[0, 30]$ with $y(0) = 1$. The first of these has a unimodal solution on the indicated interval, and we will refer to it as IVP1. The second problem is one of the test problems used by Hull *et al* [10], and we will refer to it as IVP2. These problems have solutions

$$\begin{array}{ll} \text{IVP1:} & y(x) = \frac{x}{1+x^2} \\ \text{IVP2:} & y(x) = \frac{20}{1+19e^{-x/4}} \end{array} \tag{36}$$

In Table 1 we show the results of implementing our local error control algorithm in solving both test problems. The absolute tolerance ε_A was always 10^{-10}, except for IVP1 with $\varepsilon_R = 10^{-10}$, for which $\varepsilon_A = 10^{-12}$ was used.

Table 1: Performance data for error control algorithm applied to IVP1 and IVP2

IVP1				
ε_R	10^{-4}	10^{-6}	10^{-8}	10^{-10}
RK step rejections	2	2	0	2
GL step rejections	2	5	10	19
nodes	12	20	37	79
RKGL subintervals	4	6	12	25
IVP2				
ε_R	10^{-4}	10^{-6}	10^{-8}	10^{-10}
RK step rejections	2	2	4	5
GL step rejections	2	3	5	9
nodes	10	19	39	87
RKGL subintervals	3	6	11	24

Open Science Index, Mathematical and Computational Sciences Vol:2, No:4, 2008 publications.waset.org/8341/pdf

In this table, *RK step rejections* is the number of times a smaller stepsize had to be determined at the RK nodes; *GL step rejections* is the number of times that $x_4^* \leqslant x_3$, as described in the previous section; *nodes* is the total number of nodes used on the interval of integration, including the initial node x_0; and *RKGL subintervals* is the total number of subintervals H used on the interval of integration. It is clear that as ε_R is decreased so the number of nodes and RKGL subintervals increases (consistent with a decreasing stepsize), and so there is more chance of step rejections. There are not many RK step rejections for either problem. When $\varepsilon_R = 10^{-10}$ the GL step rejections for IVP1 are 19 out a possible 25 (almost 80%), but for IVP2 the GL step rejections number only about 38%). In both cases the GL step rejections arise as a result of relatively large local error coefficients at the GL nodes, which necessarily lead to relatively small values of h, the average node separation, so that the situation $x_4^* \leqslant x_3$ is quite likely to occur.

Figures 2 and 3 show the RK5GL3 local error for IVP1 and IVP2. The curve labelled *tolerance* in each figure is $\varepsilon_R |y_i|$, which is the upper limit placed on the local relative error.

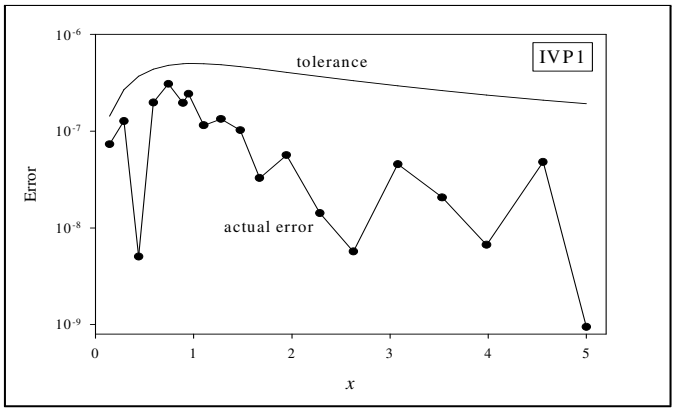


Fig. 2. RKGL local error for IVP1, with $\varepsilon_R = 10^{-6}$.

In figure 2 we have used $\varepsilon_R = 10^{-6}$, and in figure 3 we have used $\varepsilon_R = 10^{-8}$. It is clear that in both cases the tolerance has been satisfied, and the error control algorithm has been successful. In figure 4, for interest's sake, we show the stepsize variation as function of node index (#) for these two problems .

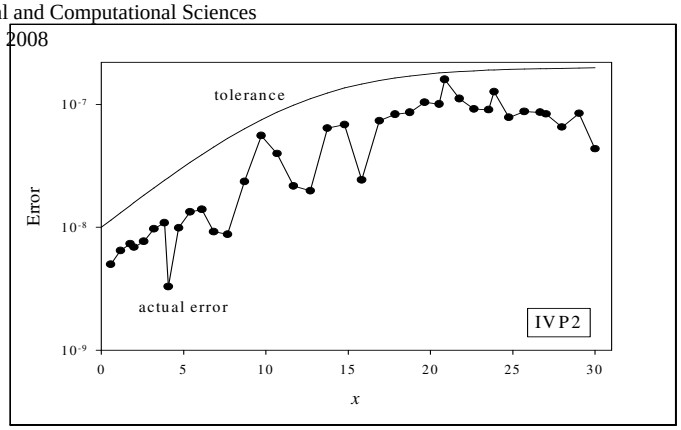


Fig. 3. RKGL local error for IVP2, with $\varepsilon_R = 10^{-8}$.

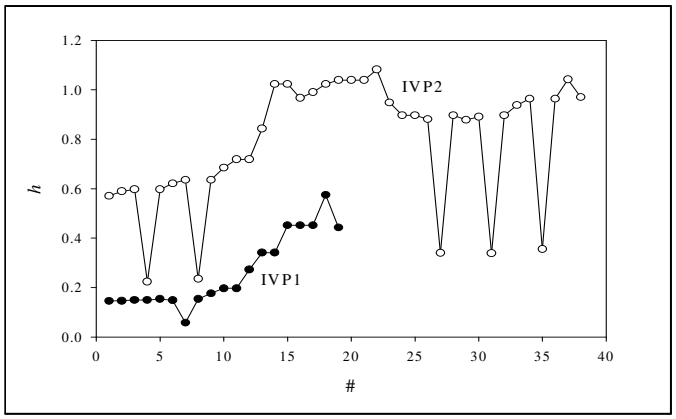


Fig. 4. Stepsize h vs node index (#) for IVP1 and IVP2.

REFERENCES

[1] J.S.C. Prentice, "The RKGL method for the numerical solution of initial-value problems", *Journal of Computational and Applied Mathematics,* 213, 2 (2008) 477.
[2] D. Kincaid and W. Cheney, *Numerical Analysis: Mathematics of Scientific Computing, 3rd ed.*, Pacific Grove: Brooks/Cole, 2002, pp492−498.
[3] E. Hairer, S.P. Norsett, and G. Wanner, *Solving ordinary differential equations I: Nonstiff problems*, Berlin: Springer-Verlag, 2000, p177.
[4] E. Hairer, S.P. Norsett, and G. Wanner, *Solving ordinary differential equations I: Nonstiff problems*, Berlin: Springer-Verlag, 2000, p180.
[5] J.C. Butcher, *Numerical methods for ordinary differential equations*, Chichester: Wiley, 2003, p192.
[6] R.L. Burden and J.D. Faires, *Numerical analysis, 7th ed.,* Pacific Grove: Brooks/Cole, 2001, pp133 − 135.
[7] E. Hairer, S.P. Norsett, and G. Wanner, *Solving ordinary differential equations I: Nonstiff problems*, Berlin: Springer-Verlag, 2000, pp165 − 185.
[8] J.C. Butcher, *Numerical methods for ordinary differential equations*, Chichester: Wiley, 2003, pp181 − 196.
[9] J.R. Dormand and P.J. Prince, "A family of embedded Runge-Kutta formulae", *Journal of Computational and Applied Mathematics*, 6 (1980) 19. Note that DOPRI853 is actually an embedded triple (8th-order, 5th-order and 3rd-order) but it is only the fifth- and eighth-order components that interest us here.
[10] T.E. Hull, W.H. Enright, B.M Fellen, and A.E. Sedgwick, "Comparing numerical methods for ordinary differential equations", *SIAM Journal of Numerical Analysis*, 9, 4 (1972) 603.

Justin Steven Calder Prentice is currently a Senior Lecturer in the Department of Applied Mathematics at the University of Johannesburg in South Africa. He holds doctoral degrees in both Physics and Applied Mathematics. The bulk of his research has been in the field of computational photovoltaics, and he has only recently turned his attention to numerical methods. His current interests are radial basis function approximation, and methods for initial value problems.